DBMS PROJECT

Topic: Design and Development of Flight Database Management System

Group Members:

- A. Lakshmana Sai Srikar (204103)
- Ch. Phani Rama Vaibhav (204115)

Problem statement:

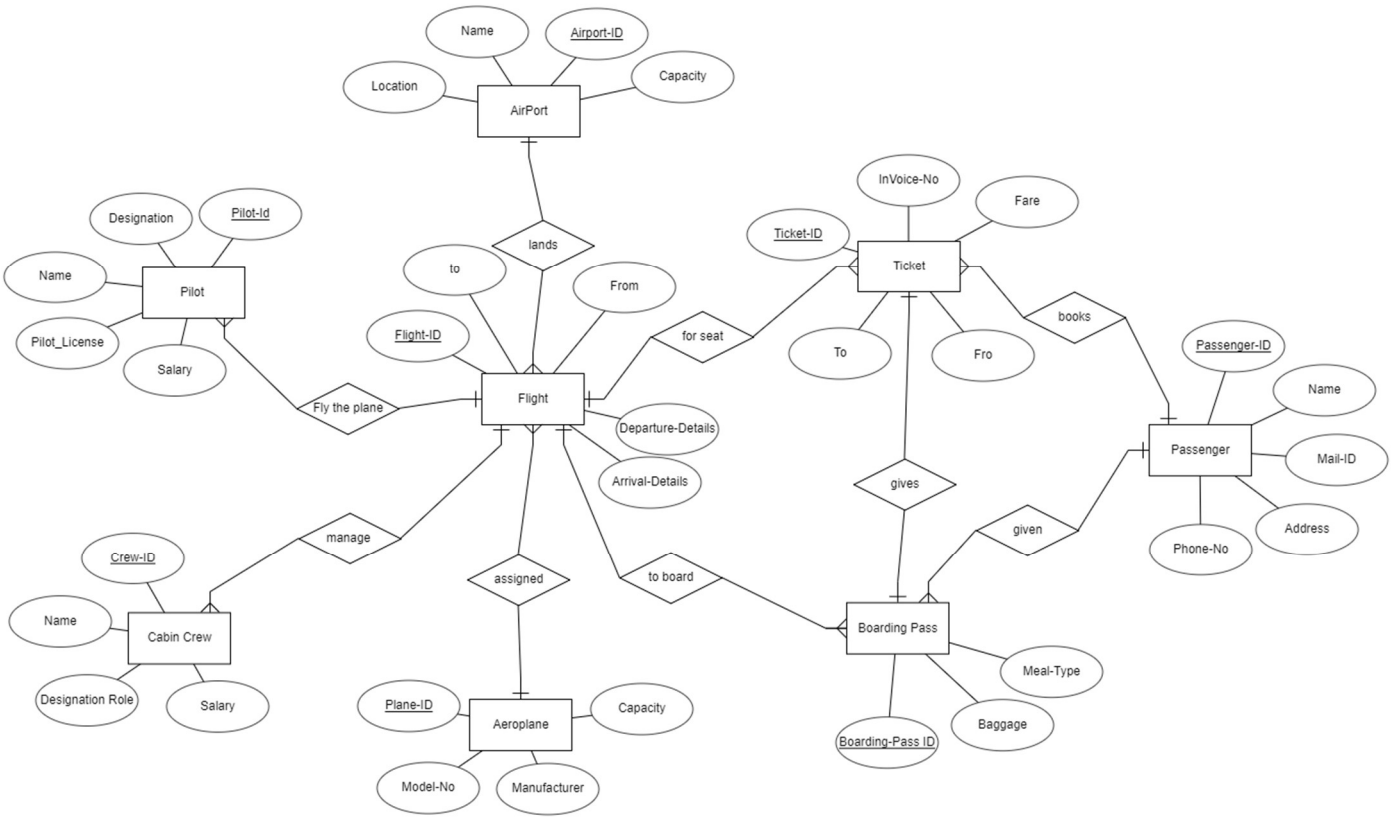
The motive in this project, is to create flight database management and booking system, this database stores the information of all the flights running between different airports, and the passenger, crew details of each flight, also the details of the airplane are stored, the features of this database are

- For booking a flight by a passenger
- An open user can access the details of flights running between two airports example :their departure and arrival times and the fare

The database also can keep track of the travel history of a passenger based on his id, this data of passenger travel preference helps in better scheduling of flights and helps to increase the flight frequency between busier airports

Contents:

ER Diagram:

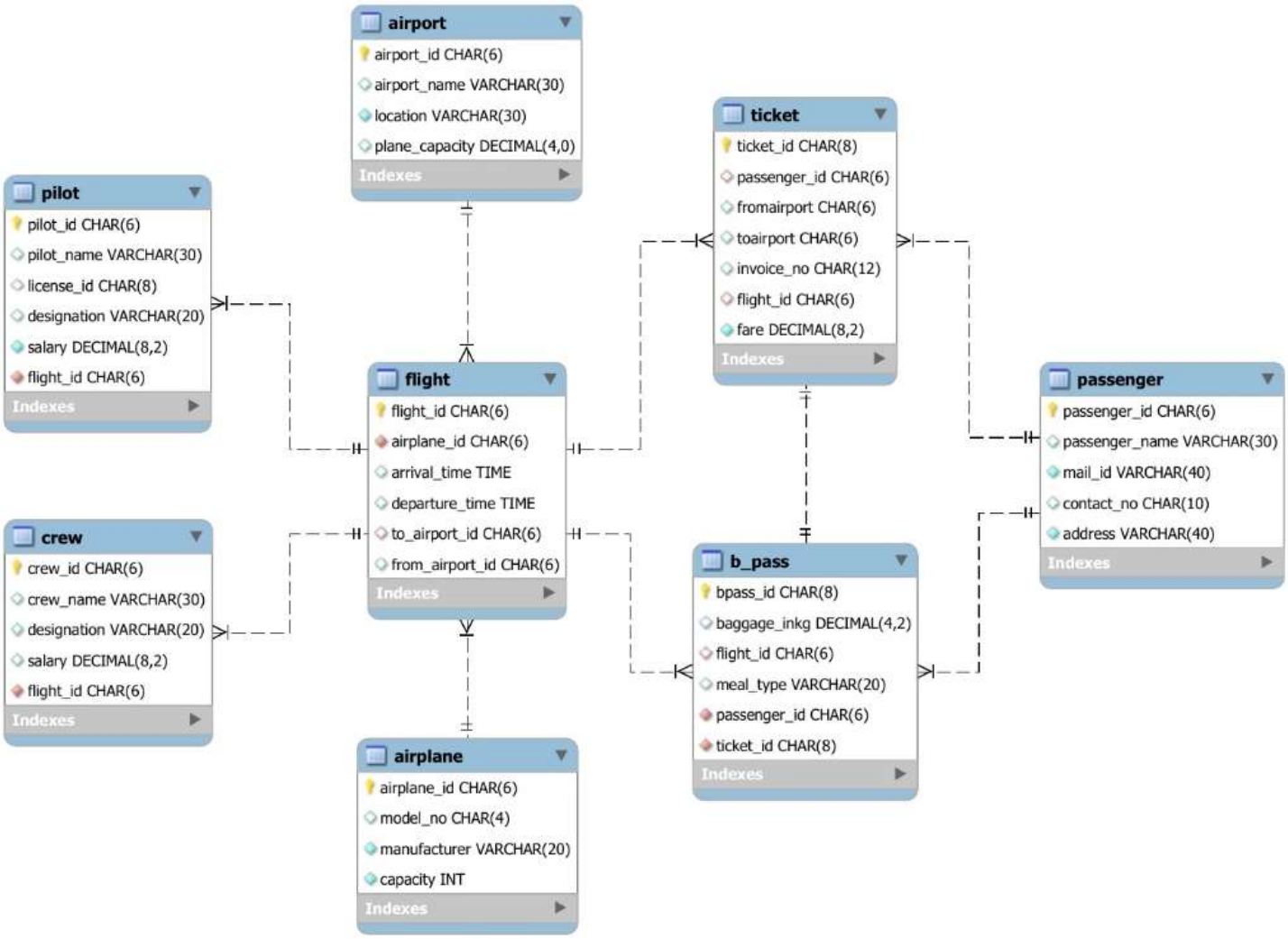


Please refer this link for a clearer image: https://drive.google.com/file/d/1i-ki5xNOyUsuotcEqdlP2crCiSDvvlsJ/view?usp=sharing

Assumptions:

- A single passenger has a single contact no and single mail id and only the permanent address is stored
- No two flights have the same flight id
- All the flights belong to a single airline (jet airways for example)
- A single airplane can have multiple flight id's
- Multiple pilots and cabin crew members are assigned to a flight
- One ticket has only one boarding pass
- There can be multiple airplanes with the same model number
  Example: there can be 4 number of Boeing 747 airplanes.

Relational Schema before normalization:



Please refer the link for a clearer image: https://drive.google.com/file/d/1Pyb-cIE_rDTiZGqHZqL9LrZmDqggStYa/view?usp=sharing

Functional Dependencies and Primary keys:

1. **Passenger**:
   passenger_id → {phone_no, name,address,mail_id }
   Since all the fields depend on the passenger_id, $(passenger\_id)^+ \rightarrow R$.
   Hence, **passenger_id** is a primary key.

2. **Airport:**
   airport_id → {name,capacity,location}
   Since all the fields depend on the airport_id, $(airport\_id)^+ \rightarrow R$.
   Hence, **airport_id** is a primary key.

3. **Cabin Crew:**
   crew_id → {name,designation} ; designation →salary ;
   flight_id →crew_id
   Since all the fields depend on the flight_id, $(flight\_id)^+ \rightarrow R$.
   Hence**, flight_id** is a primary key.

4. **Airplane:**
   airplane_id →model; model →{manufacturer,capacity}
   Since all the fields depend on the airplane_id, $(airplane\_id)^+ \rightarrow R$.
   Hence**, airplane_id** is a primary key.

5. **Boarding Pass:**
   Bpass_id →{Baggage,Flight_id,Meal_type} ;
   ticket_id →{Passenger_id,Bpass_id}
   Since all the fields depend on the ticket_id, $(ticket\_id)^+ \rightarrow R$.
   Hence, **ticket_id** is a primary key.

6. **Pilot:**
   flight_id → pilot_id;
   pilot_id →{name,pilot_license,designation}; designation →salary
   Since all the fields depend on the flight_id, $(flight\_id)^+ \rightarrow R$.
   Hence, **flight_id** is a primary key.

7. **Flight:**
   flight_id→{ arrival_time,departure_time,toairport_id,fromairport_id, airplane_id}
   Since all the fields depend on the flight_id, $(flight\_id)^+ \rightarrow R$.
   Hence, **flight_id** is a primary key.

8. **Ticket:**
   ticket_id → {passenger_id,Invoice_no,toairport,fromairport,flight_id} ;
   toairport →fare;
   fromairport →fare
   Since all the fields depend on the ticket_id, $(ticket\_id)^+ \rightarrow R$.
   Hence, **ticket_id** is a primary key.

Relations Involved:

I. **Pilot ->Flight (**Fly the plane**)**

II. **Cabin crew ->Flight (**Manage**)**

III. **Flight->Aeroplane (**Assigned**)**

IV. **Boarding pass -> Flight (**To board**)**

V. **Boarding pass -> Passenger (**Given**)**

VI. **Ticket -> Passenger (**Books**)**

VII. **Ticket -> Flight (**For seat**)**

VIII. **Flight ->Airport (**Lands**)**

IX. **Ticket -Boarding Pass** (gives)

Normalisations:

1. Passenger:

Candidate key = **passenger_id**

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

2. Airport:

Candidate key = **airport_id**

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

3. Flight:

Candidate key: **flight_id**

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

4. Cabin crew:

Candidate key= **flight_id**

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and **yes** Transitivity. Hence 3NF form is failed ✗

To convert it into stable 3NF form we need to divide it based on the relations and if suppose a relation R is split into two relations R1 &R2.
Then, it must satisfy **$(R1 \cap R2)^+ \rightarrow R1 \mid R2$**

By using that we get the relation to be split into 3 parts

- Crew: crew_id → {name,designation}

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

- Crew_salary: designation →salary

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

- Crew_Flight: flight_id →crew_id

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

5. <u>Airplane:</u>

Candidate key= **airplane_id**

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and **yes** Transitivity. Hence 3NF form is failed ✗

To convert it into stable 3NF form we need to divide it based on the relations and if suppose a relation R is split into two relations R1 &R2.
Then, it must satisfy **(R1 ∩ R2)$^+$ → R1 | R2**

By using that we get the relation to be split into 2 parts

- airplane: airplane_id →model

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

- model: model →{manufacturer,capacity}

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

6. <u>Boarding pass:</u>

Candidate Key=**ticket_id**

All the attributes are atomic. hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and **yes** Transitivity. Hence 3NF form is failed ✗

To convert it into stable 3NF form we need to divide it based on the relations and if suppose a relation R is split into two relations R1 &R2.
Then, it must satisfy **(R1 ∩ R2)$^+$ → R1 | R2**

By using that we get the relation to be split into 2 parts

- Bpass_id →{Baggage,Flight_id,Meal_type}

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

- ticket_id →{Passenger_id,Bpass_id}

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

7. <u>Pilot</u>:

Candidate key**: flight_id**

All the attributes are atomic. hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and **yes** Transitivity. Hence 3NF form is failed ✗

To convert it into stable 3NF form we need to divide it based on the relations and if suppose a relation R is split into two relations R1 &R2.
Then, it must satisfy **$(R1 \cap R2)^+ \rightarrow R1 \mid R2$**

By using that we get the relation to be split into 3 parts

- flight_id → pilot_id

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

- pilot_id →{name,pilot_license,designation}

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

- designation →salary

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

8. <u>Ticket:</u>

Candidate key**: ticket_id**

All the attributes are atomic. hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and **yes** Transitivity. Hence 3NF form is failed ✗

To convert it into stable 3NF form we need to divide it based on the relations and if suppose a relation R is split into two relations R1 &R2.
Then, it must satisfy **(R1 ∩ R2)** $^{+}$ **→ R1 | R2**

By using that we get the relation to be split into 2 parts

- fromairport →fare; toairport →fare

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

- ticket_id → {passenger_id,Invoice_no,toairport,fromairport,flight_id}

All the attributes are atomic. Hence all are in 1NF form ✓
All the attributes are in 1NF form and No Partial Dependency. Hence all are in 2NF form ✓
All the attributes are in 2NF and No Transitivity. Hence all are in 3NF form ✓

Tables (after normalization):

- All primary keys and foreign keys ate not null ,
- All the primary keys are unique

1. Passenger

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
passenger_id	char(6)	Primary key
passenger_name	varchar(30)	Not null
mail_id	varchar(40)	Not null
contact_no	char(10)	Not null
address	varchar(40)	Not null

2. Airport

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
airport_id	char(6)	primary key
airport_name	varchar(30)	Not null
location	varchar(30)	Not null
plane_capacity	numeric(4,0)	Not null

3. Model

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
model_no	char(4)	primary key
manufacturer	varchar(20)	Not null
capacity	int	Not null

4. Airplane

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
airplane_id	char(6)	primary key
model_no	char(4)	foreign key

5. Flight

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
flight_id	char(6)	primary key
airplane_id	char(6)	foreign key
arrival_time	time	Not null
departure_time	time	Not null
to_airport_id	char(6)	foreign key
from_airport_id	char(6)	foreign key

6. Fare

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
toairport	char(6)	primary key
fromairport	char(6)	primary key
fare	numeric(8,2)	Not null

7. Ticket

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
ticket_id	char(8)	primary key
passenger_id	char(6)	foreign key
fromairport	char(6)	foreign key
toairport	char(6)	foreign key
invoice_no	char(12)	Not null
flight_id	char(6)	foreign key

8. B_pass

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
bpass_id	char(8)	primary key
baggage_inkg	numeric(4,2)	Not null
flight_id	char(6)	foreign key
meal_type	varchar(20)	Not null

9. B_pass2

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
bpass_id	char(8)	foreign key
ticket_id	char(8)	foreign key
passenger_id	char(6)	foreign key

10.Pilot_salary

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
designation	varchar(20)	primary key
salary	numeric(8,2)	Not null

11.Pilot

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
pilot_id	char(6)	primary key
pilot_name	varchar(30)	Not null
license_id	char(8)	Not null
designation	varchar(20)	foreign key

12.Pilot_flight

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
pilot_id	char(6)	foreign key
flight_id	char(6)	foreign key

13.Crew_salary

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
designation	varchar(20)	primary key
salary	numeric(8,2)	Not null

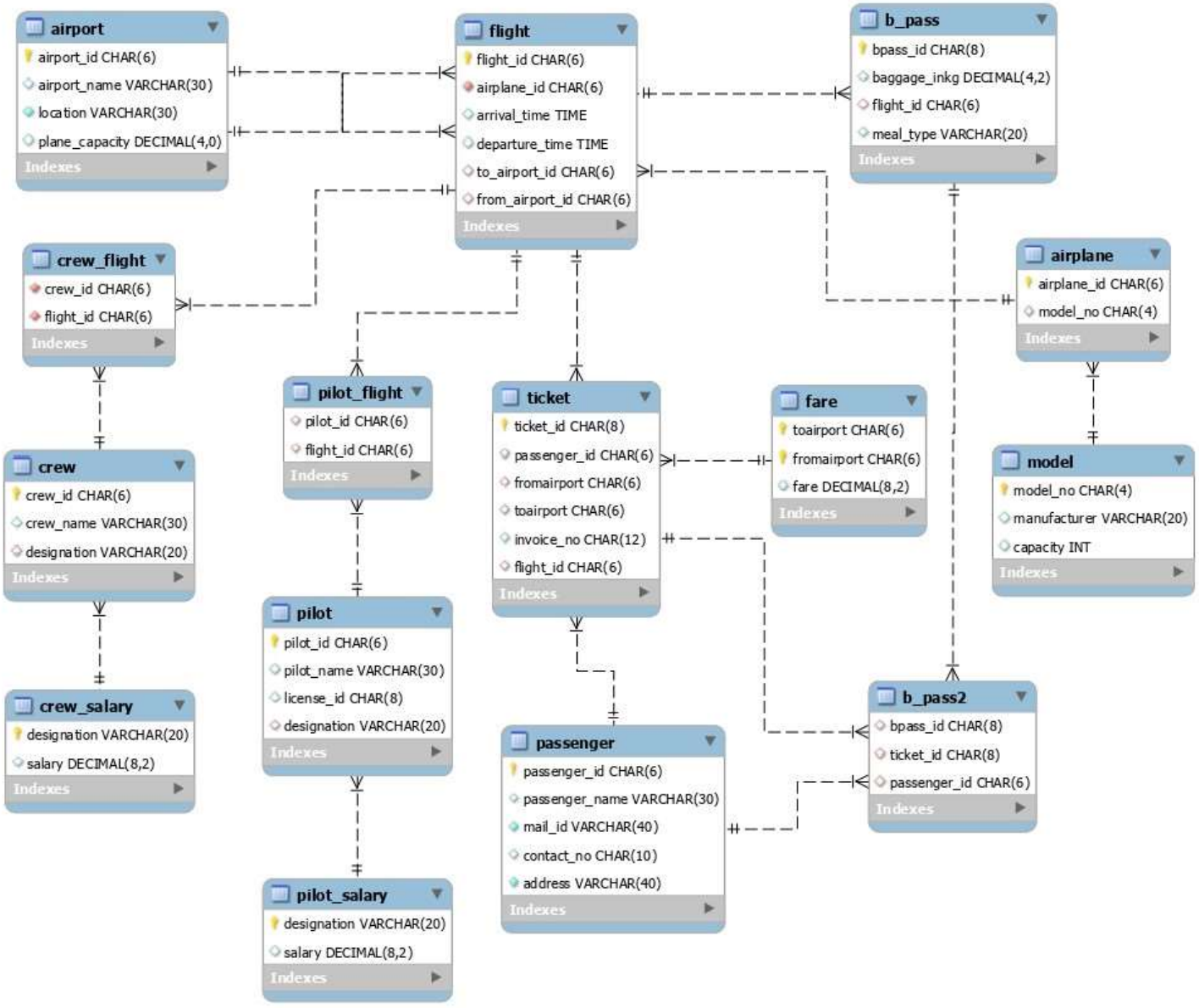
14.Crew

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
crew_id	char(6)	primary key
crew_name	varchar(30)	Not null
designation	varchar(20)	foreign key

15.Crew_flight

ATTRIBUTES	DATA TYPE	CONSTRAINT AND CHARACTERISTICS
crew_id	char(6)	foreign key
flight_id	char(6)	foreign key

Relational Schema After Normalization:



Please refer this link for clearer image:
https://drive.google.com/file/d/12LnbWf9FYOXgH2qO6lJi3002JzpV8Bu8/view?usp=sharing

SQL Code:

```
create table passenger(
passenger_id char(6),
passenger_name varchar(30),
mail_id varchar(40) not null,
contact_no char(10),
address varchar(40) not null,
primary key (passenger_id));

create table airport(
airport_id char(6),
airport_name varchar(30),
location varchar(30),
plane_capacity numeric(4,0),
primary key (airport_id));

create table model(
model_no char(4),
manufacturer varchar(20),
capacity int,
primary key (model_no));
```

```
create table airplane(
airplane_id char(6),
model_no char(4),
primary key (airplane_id),
foreign key (model_no) references model(model_no));

create table flight(
 flight_id char(6),
 airplane_id char(6),
 arrival_time time,
 departure_time time,
 to_airport_id char(6),
 from_airport_id char(6),
 primary key (flight_id),
 foreign key(airplane_id) references airplane(airplane_id),
 foreign key(to_airport_id) references airport(airport_id),
 foreign key(from_airport_id) references airport(airport_id));

create table fare(
toairport char(6),
fromairport char(6),
fare numeric(8,2),
primary key(toairport,fromairport));
```

```
create table ticket(
ticket_id char(8),
passenger_id char(6),
fromairport char(6),
toairport char(6),
invoice_no char(12),
flight_id char(6),
primary key(ticket_id),
foreign key (passenger_id) references passenger(passenger_id),
foreign key (flight_id) references flight(flight_id),
foreign key (toairport,fromairport) references fare(toairport,fromairport));

create table b_pass(
bpass_id char(8),
baggage_inkg numeric(4,2),
flight_id char(6),
meal_type varchar(20),
primary key (bpass_id),
foreign key (flight_id) references flight(flight_id));
```

```
create table b_pass2(
bpass_id char(8),
ticket_id char(8),
passenger_id char(6),
foreign key (bpass_id) references b_pass(bpass_id),
foreign key (ticket_id) references ticket(ticket_id),
foreign key (passenger_id) references passenger(passenger_id));

create table pilot_salary(
 designation varchar(20),
 salary numeric(8,2),
 primary key (designation));

create table pilot(
 pilot_id char(6),
 pilot_name varchar(30),
 license_id char(8),
 designation varchar(20),
 primary key(pilot_id),
 foreign key(designation) references pilot_salary(designation));
```

```
create table pilot_flight(
 pilot_id char(6),
 flight_id char(6),
 foreign key (pilot_id) references pilot(pilot_id),
 foreign key (flight_id) references flight(flight_id));

create table crew_salary(
 designation varchar(20),
 salary numeric(8,2),
 primary key(designation));

create table crew(
 crew_id char(6),
 crew_name varchar(30),
 designation varchar(20),
 primary key(crew_id),
 foreign key (designation) references crew_salary(designation));

create table crew_flight(
 crew_id char(6),
 flight_id char(6),
 foreign key (crew_id) references crew(crew_id),
 foreign key (flight_id) references flight(flight_id));
```

For the complete code and all the values inside of the table, please refer this link:

https://drive.google.com/file/d/1-UdL6jG5tIP6DMh6BzPlQEIZkAz3vcqh/view?usp=sharing

Checker Queries:

```
/*Query Check1*/
select passenger.passenger_id,b_pass.meal_type
from passenger,b_pass,b_pass2
where passenger.passenger_id=b_pass2.passenger_id and
b_pass2.bpass_id=b_pass.bpass_id;

/*Query Check2*/
select
airplane.airplane_id,flight.flight_id,fare.fare,model
.manufacturer
from flight,fare,airplane,model
where model.model_no=airplane.model_no and
airplane.airplane_id=flight.airplane_id and
flight.to_airport_id=fare.toairport and
flight.from_airport_id=fare.fromairport;

/*Query check 3*/
select
b_pass.flight_id,passenger.passenger_name,passenger.a
ddress
from b_pass,ticket,passenger
where b_pass.flight_id=ticket.flight_id and
ticket.passenger_id=passenger.passenger_id;

/*Query Check 4*/
select
flight.flight_id,pilot.pilot_name,pilot.designation,p
ilot_salary.salary
from flight,pilot,pilot_salary,pilot_flight
where pilot_flight.flight_id=flight.flight_id and
pilot_flight.pilot_id=pilot.pilot_id and
pilot_salary.designation=pilot.designation ;
```